



(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:  
**15.12.2021 Bulletin 2021/50**

(51) Int Cl.:  
**G06N 3/04 (2006.01) G06N 3/08 (2006.01)**

(21) Application number: **20179099.5**

(22) Date of filing: **09.06.2020**

(84) Designated Contracting States:  
**AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR**  
Designated Extension States:  
**BA ME**  
Designated Validation States:  
**KH MA MD TN**

(71) Applicants:  
• **TOYOTA JIDOSHA KABUSHIKI KAISHA**  
**TOYOTA-SHI, AICHI-KEN 471-8571 (JP)**  
• **ETH Zurich - The Swiss Federal Institute of Technology Zurich**  
**8092 Zurich (CH)**

(72) Inventors:  
• **ABBELOOS, Wim**  
**1140 Brussels (BE)**

• **OBUKHOV, Anton**  
**8092 Zurich (CH)**  
• **RAKHUBA, Maksim**  
**8092 Zurich (CH)**  
• **GEORGOULIS, Stamatios**  
**8092 Zurich (CH)**  
• **KANAKIS, Menelaos**  
**8092 Zurich (CH)**  
• **DAI, Dengxin**  
**8092 Zurich (CH)**  
• **VAN GOOL, Luc**  
**8092 Zurich (CH)**

(74) Representative: **Cabinet Beau de Loménie**  
**158, rue de l'Université**  
**75340 Paris Cedex 07 (FR)**

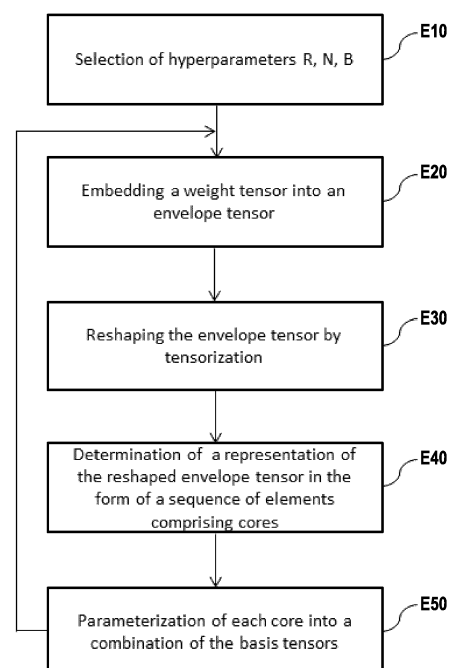
Remarks:

Amended claims in accordance with Rule 137(2) EPC.

(54) **METHOD AND SYSTEM FOR COMPRESSING A NEURAL NETWORK**

(57) A method for compressing a neural network comprising a plurality of layers, each layer comprising a weight tensor. Said method comprises a step of selecting (E10) hyperparameters to define at least one basis of tensors referred to as "basis tensors", including a rank hyperparameter R, a dimension hyperparameter N and a size hyperparameter representative of the number of basis tensors. Said method also comprises, for each weight tensor, a set of steps of:

- embedding (E20) the weight tensor into an envelope tensor,
- reshaping (E30) said envelope tensor by tensorization,
- determining (E40) a representation of the reshaped envelope tensor in the form of a sequence of elements comprising tensors referred to as "cores",
- parameterizing (E50) each core into a combination of the basis tensors.



**FIG.3**

**Description**Field of the invention

**[0001]** The present invention relates to the general field of data processing using neural networks. It relates more particularly to a method for compressing a neural network. The invention also relates to a data processing method for training such a compressed convolutional neural network. The invention finds a particularly advantageous, although not limiting, application in the case of an autonomous or semi-autonomous car.

Description of the Related Art

**[0002]** Neural networks have now become one of the most widely used approach in various fields of research to enable new technological developments, including computer vision, medical image analysis, natural language processing, etc.

**[0003]** This tremendous success is largely due to a combination of various factors, including ever deeper networks, ever larger data sets used for training, and ever more powerful computing resources (processing units) to manage such an amount of data. In this way, neural networks have gradually turned into deep networks that contain millions of trainable parameters and consume lots of memory.

**[0004]** For example, the ResNet-101 model (see reference [1]) has 44 million parameters and requires 171 MB of storage. As another example, even more resource consuming, the VGG-16 model (see reference [2]) has 138 million parameters and requires 528 MB of storage.

**[0005]** Thus, on the one hand, the ever-increasing size of neural networks implies the search for ever more advanced techniques to improve the performance of data processing managed by these neural networks. On the other hand, over the last few years, it has been witnessing a steady transition of this technology to industry. Consequently, it is becoming a pressing problem to deploy the best-performing deep networks to all kinds of resource constrained devices/systems, such as mobile robots, smartphones, wearables and IoT (acronym for "Internet of Things") devices/systems. Indeed, these devices/systems come with restrictions in terms of runtimes, latency, energy and memory consumption, which is in contrast with the considerations that the aforementioned approaches were designed with.

**[0006]** For this reason, a number of research projects are currently focusing on the possibilities of reducing the size of the neural networks to be used, while limiting the loss of performance that could be associated with such a reduction in size. It should be noted that these research projects are primarily aimed at reducing the size of a neural network that is initially large. Indeed, it was found that the parameters learned by training neural networks, in particular convolutional neural networks, were largely redundant, so that it is preferable and more efficient to train a large network and prune it, rather than directly train a smaller network.

**[0007]** Motivated by these observations, network compression has been proposed in the literature to arrive at smaller, faster, and more energy efficient neural networks. In general, network compression techniques can be grouped into the following conventional categories: pruning, hashing, quantization. More recent categories are filter/tensor decompositions which are based on a principle of low rank representation of the tensors respectively associated to the layers forming a neural network.

**[0008]** In particular, it has been proposed a Tensor Ring decomposition (see reference [3]) which offers great generalization capabilities when training compressed neural network. However, a common drawback of existing Tensor Ring approaches on network compression, like the one proposed for wide compression (see reference [4]), is that that they limit the "compression ratio - performance degradation" trade-off by having independently-compressed layer representations, they still require the implementation of high rank decompositions. Such a drawback results from the estimation of individual Tensor Ring factorizations for the tensors associated with the layers, implying an implementation that is complex and therefore costly in terms of computing resources.

**[0009]** Filter decompositions, as for them, refer to the operations of finding a low-rank (compressed) representation, which gives the smallest error of reconstructing an uncompressed representation. This class of algorithms always requires fine-tuning, and is overall very limited by the architecture.

Summary of the disclosure

**[0010]** The purpose of the present invention is to overcome all or some of the limitations of the prior art solutions, particularly those outlined here above, by providing a solution that enables to compress a neural network in a more efficient way than the solutions of the prior art.

**[0011]** To this end, and according to a first aspect, the invention relates to a method for compressing a neural network comprising a plurality of layers, each layer comprising a weight tensor whose size is equal to a product of numbers referred to as "size numbers", said method comprising a step of selecting hyperparameters to define at least one basis of tensors referred to as "basis tensors", including a rank hyperparameter R, a dimension hyperparameter N and a size

hyperparameter representative of the number of basis tensors, each basis tensor being an array belonging to a vector space whose dimension is equal to the product of at least one number belonging to the hyperparameter R and a number N' belonging to the hyperparameter N. Said method also comprises, for each weight tensor, a set of steps of:

- 5 - embedding the weight tensor into an envelope tensor, the size of the envelope tensor being equal to a product of size numbers such that:
  - if a size number M of the weight tensor is an integer power of said number N', said size number M is a size number of the envelope tensor,
  - 10 • if a size number M of the weight tensor is not an integer power of said number N', said number N' raised to a power equal to the ceiling value of log base N' of said size number M is a size number of the envelope tensor,
- reshaping said envelope tensor by tensorization based on said number N',
- determining a representation of the reshaped envelope tensor in the form of a sequence of elements comprising tensors referred to as "cores", said cores belonging to said vector space,
- 15 - parameterizing each core into a combination of the basis tensors.

[0012] The compression method according to the invention therefore proposes to parametrize each core of each of the weight tensors in said vector space, after having judiciously embedded and reshaped the weight tensor in accordance with the hyperparameters.

[0013] By "parameterization", one refers here to code instructions defining the way in which said cores are implemented in a software manner at the level of a data processing system configured to carry out said compression method, and thus a fortiori the compressed neural network resulting from such parameterization. Therefore, such a parameterization can also be seen as a computer-implemented representation defining how each core decomposes (e.g. composes) according to the tensors of the basis.

[0014] Proceeding in this way is particularly advantageous because it enables to use only one same tensor basis to express all the weight tensors of the neural network, regardless of the size of these weight tensors. In other words, the invention enables to represent a set of tensors of arbitrary shapes, all basis tensors participating in the formation of each tensor from the layers of the neural network.

[0015] Consequently, the compression method according to the invention is much more efficient than the solutions of the prior art, in particular because the inventors have shown that if D designates the number of coefficients (e.g. parameters) in a layer of the neural network, only  $O(\log_N(D))$  coefficients need to be stored to represent such tensor through given basis tensors ("O" referring to the "Big O" Bachmann-Landau notation).

[0016] In comparison, the solutions of the prior art only allow to store at least  $O(D)$  coefficients, which is higher than what the invention allows. In other words, the invention enables to compress a neural network much more efficiently.

[0017] Furthermore, using only one basis for all tensors of all layers offers the advantageous possibility to use this basis (but also coefficients of linear combinations) to compress one or more other neural networks. Thus, the tensor basis and the coefficients of the linear combinations generated form a tool that can be used to compress other neural networks efficiently and very quickly.

[0018] Finally, an additional advantage of the invention is to enable efficient computation of a product of a linear operator (such as convolutional layer or linear layer) and on operand, without the need to convert the representation (such as for example Tensor Ring representation) into full tensor representation. Instead, the product can be computed in the vector space to which the basis tensors belong.

[0019] In particular embodiments, the compression method may furthermore include one or more of the following features, taken alone or in any technically feasible combination.

[0020] According to an embodiment, for each weight tensor, the sequence associated with the reshaped envelope tensor also comprises, before each core, a diagonal matrix intended to be multiplied with said core following it in said sequence, the size of a diagonal matrix being defined as a function of the hyperparameter R.

[0021] Said diagonal matrices enables to promote extra variability in the space of tensor ring decompositions. Indeed, by introducing such diagonal matrices in the sequence of elements, extra trainable parameters are introduced between each pair of cores. Proceeding in this way increases the number of parameters to represent each layer, but enables advantageously to use a smaller basis.

[0022] According to an embodiment, for each weight tensor, the representation determined for the reshaped envelope tensor is a tensor ring decomposition.

[0023] According to an embodiment, for each weight tensor, the representation determined for the reshaped envelope tensor is a tensor train decomposition.

[0024] According to an embodiment, said at least one basis is formed of three-dimensional tensors or else four-dimensional tensors.

**[0025]** According to an embodiment, the basis tensors satisfy at least one constraint parameterized by the number of independent parameters defining each of said tensors.

**[0026]** Such features enable to represent any kind of Tensor Network decomposition.

**[0027]** According to a second aspect, the invention also relates to a data processing method. Said data processing method comprises the steps of:

- compressing a first neural network according to the invention, so as to obtain a first compressed network,
- initializing the basis tensors and the combinations from which said first compressed network is parameterized,
- training said first compressed network, the basis tensors and coefficients defining said combinations being considered as trainable parameters during said training.

**[0028]** The data processing method according to the invention inherits the advantages presented above for the compression method.

**[0029]** In particular embodiments, the data processing method may furthermore include one or more of the following features, taken alone or in any technically feasible combination.

**[0030]** According to an embodiment, said data processing method also comprises the steps of:

- compressing at least one second neural network according to the invention, so as to obtain a second compressed network, the compression of said second neural network being implemented using at least the basis tensors,
- training said second compressed network, at least coefficients defining the combinations from which said second compressed network is parameterized being considered as trainable parameters during the training of the second compressed network.

**[0031]** Thus, the tensor basis generated at the end of the training of the first compressed network are transferred to said second neural network so that it is compressed using all these parameters. In other words, the basis generated for the first compressed neural network serves as a support for the compression of said second neural network. This way of proceeding is very advantageous because it enables to limit the operating cost in terms of calculation during training and inference time.

**[0032]** Moreover, it is important to note that the invention does not impose any link between the first and second neural networks, apart from the fact that they are based on the same tensor basis. In other words, nothing excludes considering that the first and second neural networks are assigned to completely distinct tasks, such as a first VGG-type network (acronym of the expression "Visual Geometry Group") assigned to an image classification task and a second BERT-type network (acronym of the expression "Bidirectional Encoder Representations from Transformers") assigned to a text translation task.

**[0033]** According to an embodiment, the basis tensors learned during the training of the first neural network are considered as trainable parameters during the training of the second compressed network.

**[0034]** Re-training the basis tensors obtained from training the first compressed network, in addition to the coefficients of the linear combinations also obtained from training the first compressed network (and also the diagonal matrices, if any), increases the final accuracy of the second compressed network for inference.

**[0035]** Of course, the invention also covers the case where the basis tensors learned during the training of the first neural network are considered as static parameters during the training of the second compressed network.

**[0036]** According to an embodiment, when a compression step of a neural network is carried out according to the invention such that diagonal matrices are used, said diagonal matrices are also initialized during the initialization step and are also considered as learnable parameters during the training of this neural network.

**[0037]** According to a third aspect, the invention also relates to a computer program including instructions for executing the compression method according to the invention or a processing data method according to the invention when said program is executed by a computer.

**[0038]** This program can use any programming language and take the form of source code, object code or a code intermediate between source code and object code, such as a partially compiled form, or any other desirable form.

**[0039]** According to a fourth aspect, the invention also relates to a computer-readable information medium containing instructions of a computer program as described above.

**[0040]** The information medium can be any entity or device capable of storing the program. For example, the medium can include storage means such as a ROM, for example a CD ROM or a microelectronic circuit ROM, or magnetic storage means, for example a diskette (floppy disk) or a hard disk.

**[0041]** Alternatively, the information medium can be an integrated circuit in which the program is incorporated, the circuit being adapted to execute the method in question or to be used in its execution.

**[0042]** According to a fifth aspect, the invention also relates to a data processing system for compressing a neural network comprising a plurality of layers, each layer comprising a weight tensor whose size is equal to a product of

numbers referred to as "size numbers". Said processing system comprises:

- a selection module, configured to select hyperparameters to define at least one basis of tensors referred to as "basis tensors", including a rank hyperparameter R, a dimension hyperparameter N and a size hyperparameter representative of the number of basis tensors, each basis tensor being an array belonging to a vector space whose dimension is equal to the product of at least one number belonging to the hyperparameter R and a number N' belonging to the hyperparameter N,
- an embedding module, configured to embed a weight tensor into an envelope tensor, the size of the envelope tensor being equal to a product of size numbers such that:
  - if a size number M of the weight tensor is an integer power of said number N', said size number M is a size number of the envelope tensor,
  - if a size number M of the weight tensor is not an integer power of said number N', said number N' raised to a power equal to the ceiling value of log base N' of said size number M is a size number of the envelope tensor,
- a reshaping module, configured to reshape an envelope tensor by tensorization based on said number N',
- a determination module, configured to determine, for a reshaped envelope tensor, a representation of said reshaped envelope tensor in the form of a sequence of elements comprising tensors referred to as "cores", said cores belonging to said vector space,
- a parameterization module, configured to parameterize a core into a combination of the basis tensors.

**[0043]** According to a sixth aspect, the invention also relates to a system comprising:

- an initialization module, configured to initialize the basis tensors and the combinations from which a neural network is parameterized, said neural network having been compressed by means of a data processing system according to the invention,
- a training module, configured to train said compressed neural network, the basis tensors and coefficients defining said combination being considered as trainable parameters during said training.

#### Brief description of the drawings

**[0044]** How the present disclosure may be put into effect will now be described by way of example with reference to the appended drawings, in which:

- figure 1 illustrates, in its environment, a particular embodiment of a data processing system configured to compress a given neural network,
- figure 2 illustrates an example of the hardware architecture of said data processing system,
- figure 3 is a flowchart of the main steps of a compression method according to the invention, these steps being carried out by said data processing system,
- figure 4 is a flowchart of a particular embodiment of a data processing method according to the invention,
- figure 5 is a graph illustrating performance results obtained by neural networks compressed according to the invention.

#### Description of the embodiments

**[0045]** Figure 1 illustrates, in its environment, a particular embodiment of a data processing system 10 configured to compress a given neural network.

**[0046]** In a conventional way, the neural network has a structure in the form of a stack of layers. More precisely, said neural network comprises a plurality of layers, including in particular at least one convolution layer and/or one linear layer.

**[0047]** Each of these layers can be represented by means of an operator which associates a tensor X defined as an input data channel with a tensor Y defined as an output data channel, by means of the following equation:

$$Y = W X + V,$$

where W represents a weight tensor and V represents a bias vector.

**[0048]** In other words, each layer of the neural network comprises, inter alia, a weight tensor.

**[0049]** For the rest of the description, it is considered in a non-limiting way that the neural network is a convolutional neural network comprising L layers, L being a natural number strictly greater than 1. Moreover, one adopts the notation

according to which a weight tensor of layer  $l$ ,  $l$  being between 1 and  $L$ , is referred to as  $W_l$ . In a similar manner, input data channel and output data channel for said layer  $l$  are respectively referred to as  $X_l$  and  $Y_l$ .

**[0050]** However, it should be noted that the invention remains applicable whatever the nature of the neural network considered, in particular for any deep neural network.

**[0051]** It also should be noted that no limitation is attached to the kind of digital data that may be received through the input data channel of the first layer of the convolutional neural network. In the same way, no limitation is attached to the kind of score or classification that may be output based on the input.

**[0052]** For example, if the inputs to the convolutional neural network are images or features that have been extracted from images, the output generated by the convolutional neural network for a given image may be scores for each of a set of object categories, with each score representing an estimated likelihood that the image contains an image of an object belonging to the category.

**[0053]** According to another example, if the inputs to the convolutional neural network are Internet resources (e.g., web pages), documents, or portions of documents or features extracted from Internet resources, documents, or portions of documents, the output generated by the convolutional neural network for a given Internet resource, document, or portion of a document may be a score for each of a set of topics, with each score representing an estimated likelihood that the Internet resource, document, or document portion is about the topic.

**[0054]** According to another example, if the inputs to the convolutional neural network are features of an impression context for a particular advertisement, the output generated by the convolutional neural network may be a score that represents an estimated likelihood that the particular advertisement will be clicked on.

**[0055]** According to another example, if the inputs to the convolutional neural network are features of a personalized recommendation for a user, e.g., features characterizing the context for the recommendation, e.g., features characterizing previous actions taken by the user, the output generated by the convolutional neural network may be a score for each of a set of content items, with each score representing an estimated likelihood that the user will respond favourably to being recommended the content item.

**[0056]** According to another example, if the input to the convolutional neural network is text in one language, the output generated by the convolutional neural network may be a score for each of a set of pieces of text in another language, with each score representing an estimated likelihood that the piece of text in the other language is a proper translation of the input text into the other language.

**[0057]** According to another example, if the input to the convolutional neural network is a spoken utterance, a sequence of spoken utterances, or features derived from one of the two, the output generated by the convolutional neural network may be a score for each of a set of pieces of text, each score representing an estimated likelihood that the piece of text is the correct transcript for the utterance or sequence of utterances.

**[0058]** The size of a weight tensor is equal to a product of numbers referred to as "size numbers" (also called "dimensions" in the literature of the technical field of the invention). In a manner known per se, said size numbers comprise numbers whose respective size are equal to the sizes of the input and output data channels associated with the layer to which said weight tensor belongs.

**[0059]** For example, considering a two-dimensional convolutional layer of the convolutional neural network, the size of the input data channel of said convolutional layer is equal to:

$$S_w \times S_h \times C_{in}$$

where  $S_w$ ,  $S_h$ ,  $C_{in}$  refer respectively to the width, the height and the number of parameters (i.e. features) of said input data channel. The size of the output data channel of said convolutional layer is, as for it, equal to:

$$(S_w - K + 1) \times (S_h - K + 1) \times C_{out}$$

where  $K$  and  $C_{out}$  refer respectively to the filter size (i.e. the size of the filter which is used for said convolutional layer to extract features of the input data channel) and the number of parameters (i.e. features) of said output data channel. Thus, the size of the weight tensor associated to the conventional layer of said example is equal to:

$$C_{out} \times C_{in} \times K \times K.$$

**[0060]** According to another example, considering a linear layer of the convolutional neural network, the sizes of the input data channel and output data channel are respectively equal to  $C_{in}$  and  $C_{out}$ . Thus, the size of the weight tensor associated to the linear layer of said example is equal to  $C_{out} \times C_{in}$ .

**[0061]** It should be noted that with respect to the use of a convolution neural network, the invention is of course not limited to a two-dimensional network, and can be applied to other networks, such as three-dimensional convolution networks.

**[0062]** The data processing system 10 is configured to carry out processes aimed at reducing the number of trainable parameters that should be learned, by implementing a method for compressing said convolutional neural network.

**[0063]** Figure 2 illustrates an example of the hardware architecture of the data processing system 10 for the implementation of the compression method according to the invention.

**[0064]** To this end, the data processing system 10 has the hardware architecture of a computer. As shown in Figure 2, the data processing system 10 comprises, in particular, a processor 1, a random access memory 2, a read-only memory 3, a non-volatile memory 4 and communication means 5.

**[0065]** The read-only memory 3 of the data processing system 10 constitutes a recording medium conforming to the invention, which is readable by processor 1 and on which is recorded a computer program PROG conforming to the invention, containing instructions for carrying out the steps of the compression method according to the invention. The program PROG defines functional modules of the data processing system 10, which are based on or control the aforementioned hardware elements 2 to 5 of the data processing system 10, and which comprise in particular:

- a selection module MOD\_SEL, configured to select hyperparameters to define at least one basis of tensors referred to as "basis tensors", including a rank hyperparameter R, a dimension hyperparameter N and a size hyperparameter B representative of the number of basis tensors, each basis tensor being an array belonging to a vector space V whose dimension is equal to the product of at least one number belonging to the hyperparameter R and a number N' belonging to the hyperparameter N,
- an embedding module MOD\_EMB, configured to embed a weight tensor into an envelope tensor, the size of the envelope tensor being equal to a product of size numbers such that:
  - if a size number M of the weight tensor is an integer power of said number N', said size number M is a size number of the envelope tensor,
  - if a size number M of the weight tensor is not an integer power of said number N', said number N' raised to a power equal to the ceiling value of log base N' of said size number M is a size number of the envelope tensor,
- a reshaping module MOD\_RSHA, configured to reshape an envelope tensor by tensorization based on said number N',
- a determination module MOD\_DET, configured to determine, for a reshaped envelope tensor, a representation of said reshaped envelope tensor in the form of a sequence of elements comprising tensors referred to as "cores", said cores belonging to said vector space V,
- a parameterization module MOD\_PAR, configured to parameterize a core into a combination of the basis tensors.

**[0066]** The functionalities attached to each of the functional modules are explained in detail later on when describing modes of implementation of said compression method.

**[0067]** In the present embodiment, it is considered that the data processing system 10 takes the form of a single computer comprising all the functional modules mentioned above.

**[0068]** However, nothing excludes the possibility of considering that the data processing system 10 comprises a plurality of computers configured in a similar manner to said single computer, said functional modules being distributed between the computers of said plurality of computers (in which case, the PROG program comprises a plurality of sub-programs also distributed between said computers of said plurality of computers).

**[0069]** Moreover, in the present embodiment, it is also considered that the convolutional neural network (i.e. the code instructions defining said convolutional neural network) is previously stored in a memory of the data processing system 10, such as the non-volatile memory 4.

**[0070]** In an alternative embodiment, said convolutional neural network is initially stored in a memory equipping an entity external to the data processing system 10 (e.g.: database server). Therefore, in this alternative embodiment, the data processing system 10 also includes an obtaining module configured to obtain said convolutional neural network. In other words, said convolutional neural network (i.e. the code instructions defining said convolutional neural network) is obtained via a data exchange (transmission/reception) controlled by the obtaining module and carried out by the communication means 5 of said data processing system 10 as well as by communication means suitable for transmission equipping the external entity.

**[0071]** In general, such an exchange of data between the obtaining module and the said entity is carried out via a communication interface. No limitation is attached to the nature of this communication interface, which may be wired or non-wired, and may implement any protocol known to the person skilled in the art (Ethernet, Wifi, Bluetooth, 3G, 4G, 5G, etc.), the communication means 5 of the data processing system 10 being also configured accordingly.

**[0072]** More generally, no limitation is attached to the way in which the convolutional neural network is obtained by the data processing system 10.

**[0073]** The hyperparameters R, N and B are vectors each comprising natural numbers from which said at least one basis of tensors is defined according to the invention.

5 **[0074]** Some notations are now briefly introduced here in order to remind the definitions of the hyperparameters R and N, these definitions being well known to the person skilled in the art.

**[0075]** Let A be a d-dimensional tensor:

$$10 \quad A = \mathbb{R}^{N_1 \times \dots \times N_d}$$

**[0076]** Therefore,  $A(i_1, i_2, \dots, i_d)$  represents the element of tensor A which is in the position  $(i_1, i_2, \dots, i_d)$ . Said position  $(i_1, i_2, \dots, i_d)$  belongs to a set I:

$$15 \quad I = I_1 \times \dots \times I_d$$

where:

$$20 \quad I_k = \{1, \dots, N_k\}$$

**[0077]** The element  $(N_1, \dots, N_d)$  corresponds to the dimension hyperparameter N mentioned above.

25 **[0078]** Several types of representations exist for such a tensor A. For example, such a representation is the Tensor Ring decomposition according to which:

$$30 \quad A(i_1, \dots, i_d) = \sum_{\substack{r_1, \dots, r_d=1 \\ r_{d+1}=r_1}}^{R_1, \dots, R_d} \prod_{k=1}^d C_k(r_k, i_k, r_{k+1})$$

35 where:

$$40 \quad C_k \in \mathbb{R}^{R_k \times N_k \times R_{k+1}}, r_k \in \{1, \dots, R_k\}$$

**[0079]** The element  $(R_1, \dots, R_d)$  is conventionally referred to as the Tensor Ring rank, and corresponds to the hyperparameter R mentioned above. Moreover, each  $C_k$  corresponds to said core which is a three dimensional tensor.

45 **[0080]** It should be noted that representations other than the Tensor-Ring representation may be considered, such as for example the Tensor-Train decomposition which actually corresponds to the case where  $R_1$  is equal to 1 in the above mentioned Tensor-Ring decomposition.

**[0081]** Figure 3 is a flowchart of the main steps of the compression method according to the invention, these steps being carried out by the data processing system 10.

50 **[0082]** For the description of said compression method, and for the sake of simplicity, it is considered in a non-limiting way that the hyperparameter R (respectively the hyperparameter N) associated with any one of the weight tensors of a layer I is a vector whose components are all equal to the same natural number R' (respectively the same natural number that is still denoted N' in the following). In this way, only one basis of tensors is considered for the compression method of figure 3.

55 **[0083]** Furthermore, it is also considered in a non-limiting way that the expression "representation of a tensor" refers to a Tensor Ring decomposition. The way in which the compression method can be considered for a different representation from that of Tensor Ring is described later.

**[0084]** As shown in Figure 3, the compression method comprises a step E10 of selecting the hyperparameter R, N and B.

**[0085]** This selection step E10 is performed by the selection module MOD\_SEL of the data processing system 10.



[0086] By way of example, the number R' is selected in the set {2,...,32}, the number N' is selected in the set {2, 3, 4, 9} (preferably matching the size of a spatial factor of the convolutional kernel of a convolutional layer), and the hyperparameter B is selected in the set {1,...,128}.

[0087] Obviously, nothing excludes considering, following other examples not detailed here, other sets within which the hyperparameters R, N and B are selected.

[0088] Once said hyperparameters have been selected, the compression method comprises a set S<sub>E</sub> of steps which are iterated for each of the weight tensors W<sub>I</sub> belonging to the convolutional neural network.

[0089] As shown in Figure 3, said set S<sub>E</sub> of steps comprises, for a given weight tensor W<sub>I</sub> (i.e. index I is fixed), a step E20 of embedding the weight tensor W<sub>I</sub> into an envelope tensor  $\mathbb{W}_I$ .

[0090] This embedding step E20 is performed by the embedding module MOD\_EMB of the data processing system 10.

[0091] Said embedding step E20 is performed such that the size of the envelope tensor  $\mathbb{W}_I$  is equal to a product of size numbers. More precisely, said size numbers are defined such that:

- if a size number M of the weight tensor W<sub>I</sub> is an integer power of said number N', said size number M is a size number of the envelope tensor  $\mathbb{W}_I$ ,
- if a size number M of the weight tensor W<sub>I</sub> is not an integer power of said number N', said number N' raised to a power equal to the ceiling value of log base N' of said size number M is a size number of the envelope tensor.

[0092] Thus, the envelope tensor  $\mathbb{W}_I$  is a tensor enclosing the weight tensor W<sub>I</sub> in the sense that the size numbers that define its dimension are greater or equal than the corresponding size numbers of the weight tensor W<sub>I</sub>.

[0093] It should be noted that the principle of embedding a weight tensor into an envelope tensor is known to the man skilled in the art, and is mostly based on natural integer factorization techniques. By way of example, it is possible to refer to reference [5] for more details.

[0094] In order to illustrate the operation of said step E30, let's consider, according to a first example, a number N' equal to 2, as well as a weight tensor W<sub>I</sub> of size 32x16x3x3 and belonging to a convolution layer of the convolutional neural network. Each of the numbers 32, 16, and 3 corresponds to a size number of said weight tensor W<sub>I</sub>. Specifically, the numbers 32 and 16 correspond respectively to the number Cout of parameters of the data output channel and the number Cin of parameters of the data input channel for said convolution layer. The number 3, as for it, corresponds to the size K of the filter used for said convolution layer.

[0095] Therefore, let us first consider the size numbers 32 and 16. These numbers are integer powers of the number N' (32=2<sup>5</sup> and 16=2<sup>4</sup>). Therefore, according to the invention, these numbers 32 and 16 are also size numbers of the envelope tensor.

[0096] The size number 3, on the other hand, is not an integer power of the number N'. Therefore, in step E30, the following quantity is calculated:

$$\lceil \log_{N'}(3) \rceil = \lceil \log_2(3) \rceil = 2$$

The result of this calculation is that the number N' raised to the power 2 corresponds to a size number of the envelope tensor  $\mathbb{W}_I$ , in accordance with the number 3 of the weight tensor W<sub>I</sub>. In the end, the envelope tensor  $\mathbb{W}_I$  associated with the weight tensor W<sub>I</sub> has a size of 32x16x4x4.

[0097] It should be noted that the size of the envelope tensor  $\mathbb{W}_I$  is different from 32x16x4x4 if the number N' is different from 2. Indeed, let's consider, according to a second example, a number N' equal to 3 for a weight tensor W<sub>I</sub> whose size is again equal to 32x16x3x3. The size number 32 is not an integer power of the number N'. Therefore, by performing calculations similar to those performed previously, the following quantity

$$\lceil \log_{N'}(32) \rceil = \lceil \log_3(32) \rceil = 4$$

is calculated in step E30. The result of this calculation is that the number N' raised to the power 4 corresponds to a size number of the envelope tensor  $\mathbb{W}_I$ , in accordance with the number 32 of the weight tensor W<sub>I</sub>. In the end, after all the calculations are done, the envelope tensor  $\mathbb{W}_I$  associated with the weight tensor W<sub>I</sub> has a size of 81x27x3x3.

[0098] Said set S<sub>E</sub> of steps also comprises, for said given weight tensor W<sub>I</sub>, a step E30 of reshaping said envelope tensor  $\mathbb{W}_I$  by tensorization.

[0099] This reshaping step E30 is performed by the reshaping module MOD\_RSHA of the data processing system 10.  
 [0100] It should be noted that the principle of reshaping a tensor by tensorization is known to the man skilled in the art, and is mostly based on generating "artificial" dimensions by factorising each of the size numbers of a tensor by a given integer. By way of example, it is possible to refer to reference [6] for more details.

[0101] In the present embodiment, the aim of the reshaping according to step E30 is to factorize the size numbers of the envelope tensor  $\hat{W}_l$  by the number  $N'$ , so as to generate said artificial dimensions.

[0102] In order to illustrate the implementation of step E30, let's consider the previous example where the weight tensor  $W_l$  has a size of  $32 \times 16 \times 3 \times 3$ , and where  $N'$  is equal to 2 so that the envelope tensor  $\hat{W}_l$  has a size of  $32 \times 16 \times 4 \times 4$ . Thus, at the end of step E30, a reshaped envelope tensor  $\hat{W}_l$  is obtained whose size is equal to:

$$(2 \times 2 \times 2 \times 2 \times 2) \times (2 \times 2 \times 2 \times 2) \times (2 \times 2) \times (2 \times 2)$$

[0103] In the latter expression, the first five occurrences of the number  $N'$  correspond to the decomposition of the size number 32 of the envelope tensor  $\hat{W}_l$  as a function of said number  $N'$ , the next four occurrences of the number  $N'$  correspond to the decomposition of the size number 16 of the envelope tensor  $\hat{W}_l$  as a function of said number  $N'$ , etc.

[0104] Said set  $S_E$  of steps also comprises, for said given weight tensor  $W_l$ , a step E40 of determining a Tensor-Ring decomposition of the reshaped envelope tensor  $\hat{W}_l$ .

[0105] This determination step E40 is performed by the determination module MOD\_DET of the data processing system 10.

[0106] Said Tensor-Ring decomposition is determined in the form of a sequence of elements comprising said cores previously mentioned, said cores belonging to the vector space  $V$  generated by the basis tensors as described in more detail below. Otherwise said, the reshaped envelope tensor  $\hat{W}_l$  may be written as a combination of cores referred to as  $C_k^l$ , the index  $l$  referring to the fact that this core is determined for the layer  $l$  of the convolutional neural network.

[0107] According to a particular embodiment, each element of said sequence is a core of the Tensor-Ring decomposition. Therefore, according to this embodiment, the reshape envelope tensor  $\hat{W}_l$  may be written:

$$\hat{W}_l = \sum_{\substack{r_1, \dots, r_d=1 \\ r_{d+1}=r_1}}^{R', \dots, R'} \prod_{k=1}^{d_l} C_k^l(r_k, i_k, r_{k+1})$$

where  $d_l$  is the dimension of the reshaped envelope tensor  $\hat{W}_l$ , i.e. the number of occurrences of the number  $N'$  in writing the size of said reshaped envelope tensor after reshaping by tensorization.

[0108] For example, let's consider the previous example where the size of the reshaped envelope tensor  $\hat{W}_l$  is equal to:

$$(2 \times 2 \times 2 \times 2 \times 2) \times (2 \times 2 \times 2 \times 2) \times (2 \times 2) \times (2 \times 2)$$

Thus, according to this example, the reshaped envelope tensor  $\hat{W}_l$  admits a Tensor-Ring decomposition in which thirteen (i.e.  $5 + 4 + 2 + 2$ ) cores  $C_k^l$  are used, each core  $C_k^l$  being a three dimensional tensor of the shape  $R' \times 2 \times R'$ .

[0109] In another particular embodiment, the sequence associated with the reshaped envelope tensor  $\hat{W}_l$  also comprises, before each core  $C_k^l$ , a diagonal matrix  $\Omega_k^l$  intended to be multiplied with said core  $C_k^l$  following it in said sequence. The size of the diagonal matrix  $\Omega_k^l$  is defined as a function of the hyperparameter  $R$ . More precisely, considering here that the hyperparameter  $R$  is a vector whose components are all equal to the same natural number  $R'$ , the diagonal matrix  $\Omega_k^l$  is defined with:

$$\Omega_k^l \in \mathbb{R}^{R' \times R'}$$

Consequently, in this particular embodiment, the reshape envelope tensor  $\hat{W}_l$  may be written:

$$\hat{W}_l = \sum_{\substack{r_1, \dots, r_d=1 \\ r_{d+1}=r_1}}^{R', \dots, R'} \prod_{k=1}^{d_l} \Omega_k^l(r_k, r_k) C_k^l(r_k, i_k, r_{k+1})$$

where  $\Omega_k^l(r_k, r_k)$  corresponds to the element of the diagonal of  $\Omega_k^l$  placed in position  $(r_k, r_k)$ .

**[0110]** Said diagonal matrices enables to promote extra variability in the space of tensor ring decompositions. Indeed, by introducing such diagonal matrices in the sequence of elements,  $R'$  extra trainable parameters are introduced between each pair of cores. Proceeding in this way increases the number of parameters to represent each layer, but enables advantageously to use a smaller basis.

**[0111]** It should be noted that the inventors have found, following numerical simulations, that the best results in terms of accuracy for classification and segmentation are obtained when the values of the diagonal matrices are kept positive.

**[0112]** Said set  $S_E$  of steps also comprises, for said given weight tensor  $W_l$ , a step E50 of parameterizing each core associated to the reshaped envelope tensor  $\hat{W}_l$  into a combination of the basis tensors.

**[0113]** This decomposition step E50 is performed by the parameterization module MOD\_PAR of the data processing system 10.

**[0114]** By "parameterization", one refers here to code instructions defining the way in which said cores are implemented in a software manner at the level of the data processing system 10, and thus a fortiori the compressed neural network resulting from such parameterization. Therefore, such a parameterization can also be seen as a computer-implemented representation defining how each core decomposes (e.g. composes) according to the tensors of the basis.

**[0115]** In other words, according to said step E50, it is imposed that all the cores  $C_k^l$  belong to said vector space  $V$  generated by the basis tensors. Thus, if the basis tensors are referred as  $\beta_b$  with:

$$\beta_b \in \mathbb{R}^{R' \times N' \times R'}, \quad b \in \{1, \dots, B\}$$

said vector space  $V$  may be defined by:

$$V \stackrel{\text{def}}{=} \left\{ \sum_{b=1}^B \delta_b \beta_b \mid \delta_b \in \mathbb{R} \right\} \subseteq \mathbb{R}^{R' \times N' \times R'}$$

**[0116]** As a result, according to the present embodiment, each core  $C_k^l$  of the reshaped envelope tensor  $\hat{W}_l$  may be written:

$$C_k^l = \sum_{b=1}^B \alpha_{k,b}^l \beta_b, \quad k \in \{1, \dots, d_l\}$$

where  $\alpha_{k,b}^l$  are the coefficients of said linear combination from which the compressed neural network is parameterized.

**[0117]** It should be noted that the choice of having each core decompose into a sum of basis tensors is only a variant of implementation of the invention. The invention remains indeed applicable for any type of parameterization. Thus, the aforementioned sum can be replaced, for example, by a product between the different tensors of the base, or, more generally, by any logical operation between these tensors.

**[0118]** The set  $S_E$  of steps E20 to E50 has so far been described in relation to a single weight tensor  $W_I$  of the convolutional neural network. However, it should be noted that in accordance with the invention, said set  $S_E$  of steps E20 to E50 is iterated for each of the weight tensors contained in said convolutional neural network.

**[0119]** Decomposing each core of each of the weight tensors in the vector space  $V$  is particularly advantageous because it enables to use only one same tensor base to express all the weight tensors of the convolutional neural network, regardless of the size of these weight tensors. Proceeding in this way is much more efficient than the solutions of the prior art, in particular because the inventors have shown that if  $D$  designates the number of coefficients (e.g. parameters) in a layer of the convolutional neural network, only  $O(\log_N(D))$  coefficients need to be stored in the case where no diagonal matrices  $\Omega_k^l$  are considered, and only  $O(L \times \log_N(D))$  coefficients need to be stored to represent such tensor through given basis tensors ("O" referring to the "Big O" Bachmann-Landau notation).

**[0120]** In comparison, the solutions of the prior art only allow to store at least  $O(D)$  coefficients, which is higher than what the invention allows. In other words, the invention enables to compress a convolutional neural network much more efficiently.

**[0121]** The compression method has been described so far considering that the hyperparameter  $R/N$  is a vector whose components are all equal to the same natural number  $R'/N'$ , so that only one basis of tensors is considered for the compression method of figure 3. However, the invention still applies in cases where several bases of tensors are considered.

**[0122]** For example, it is possible to consider bases of tensors to represent cores of the shape  $R_1 \times N' \times R_2$ ,  $R_2 \times N' \times R_3$ , etc. This is particularly advantageous when the representation determined for the reshaped envelope tensor is a Tensor-Train decomposition.

**[0123]** In addition, and according to another example, it is also possible to consider several bases of tensors for representing cores of the form  $R' \times N_1 \times R'$ ,  $R' \times N_2 \times R'$ , etc.

**[0124]** More generally, both previous examples can be combined, the compression method according to the invention thus enabling to take into consideration any core of the form  $R_i \times N_j \times R_k$  where  $R_i$  and  $R_j$  belong to  $R$  and  $N_j$  belongs to  $N$ .

**[0125]** The compression method has also been described so far considering that the basis tensors were three-dimensional tensors. However, nothing excludes considering one or more bases composed of four-dimensional tensors. In this way, it is possible to consider MERA (acronym of the expression "Multiscale Entanglement Renormalization Ansatz") decomposition for the weight tensors of the convolutional neural network.

**[0126]** Moreover, the compression method has also been described so far considering that the basis tensors are unconstrained, meaning they are represented with  $R \times N \times R$  parameters, these parameters being trainable. However, nothing excludes considering that the basis tensors satisfy at least one constraint parameterized by the number of independent parameters defining each of said tensors.

**[0127]** For example, such a constraint corresponds to the fact of imposing on the basis tensors to belong to a two dimensional Stiefel frame of size  $R \times (N \times R)$  or  $(R \times N) \times R$ . In particular, such an example of constraint is particularly suitable when the representation determined for the reshaped envelope tensor is a Tensor-Train decomposition.

**[0128]** Furthermore, it should be noted that the invention is not limited to a method for compressing a convolutional neural network. Indeed, when a convolutional neural network is compressed according to said compression method, it can then be used, in accordance with another aspect of the invention, by a system configured to carry out processes aimed at initializing and training said network, by implementing a data processing method according to the invention.

**[0129]** For the sake of simplicity, and for the rest of the description, it is considered in a non-limiting way that the system configured to initialize and to train the compressed convolutional neural network is said data processing system 10. However, nothing excludes considering that the compressed convolutional neural network is first obtained by said data processing system 10 and then transmitted to another system for initializing and training. In this case, said other system is equipped with communication means compatible with the communication means 5 equipping the data processing system 10, and comprises an obtaining module controlling said communication means to obtain said compressed neural network.

**[0130]** As shown in Figure 2, the functional modules defined by the program PROG of said data processing system 10 also comprise:

- an initialization module MOD\_INI, configured to initialize the basis tensors and the combination from which a convolutional neural network is parameterized by means of the data processing system 10,
- a training module MOD\_TRAIN, configured to train said convolutional neural network, the basis tensors and coefficients defining said combination being considered as trainable parameters during said training.

**[0131]** Figure 4 is a flowchart of a particular embodiment, carried out by the data processing system 10, of the data processing method according to the invention.

**[0132]** As shown in Figure 4, the data processing method comprises a first step F10 of compressing a first convolutional neural network, so as to obtain a first compressed network.

[0133] This step F10 is performed by the data processing system 10 according to the compression method described above.

[0134] In order to train said first compressed network, it is of course important that the basis tensors and the combinations from which said first compressed network is parameterized (but also the diagonal matrices if any) are initialized. To this end, the data processing method also includes a step F20 dedicated to that purpose.

[0135] This initialization step F20 is performed by the initialization module MOD\_INI of the data processing system 10.

[0136] It should be noted that proper coefficient initialization plays an important role in convergence rate and quality of discovered local minima when the quality of the output of a compressed neural network is evaluated against output obtained without compression.

[0137] Thus, according to a particular embodiment, the initialization step F20 is performed following the practices described in reference [7]. General principle of these practices is described here. They consist in initializing each coefficient of a reshaped weight tensor  $\hat{W}_l$  by sampling it as an i.i.d. (acronym of the expression "independent and identically distributed") random variable from a normal distribution  $\mathcal{N}(0, \sigma_W^2)$ , where  $\sigma_W^2$  is determined from the shape of the

weight tensor. As previously described, the reshaped weight tensor  $\hat{W}_l$  is represented with internal coefficients  $\alpha_{k,b}^l$

and basis tensors  $\beta_b$  shared between all layers (and also  $\Omega_k^l$ , if applicable). Let's assume first that the coefficients  $\alpha_{k,b}^l$  and the basis tensors  $\beta_b$  are i.i.d. random variables sampled from normal distribution  $\mathcal{N}(0, \sigma_b^2)$  and  $\mathcal{N}(0, \sigma_\alpha^2)$  respectively, and that the diagonal matrices  $\Omega_k^l$  are not taken into account. Therefore, variance of each core  $C_k^l$  is given as follows:

$$Var(C_k^l) = Var\left(\sum_{b=1}^B \alpha_{k,b}^l \beta_b\right) = B \times \sigma_B^2 \times \sigma_\alpha^2$$

and consequently:

$$\begin{aligned} Var(\hat{W}_l(i_1, \dots, i_{d_l})) \\ &= Var\left(\sum_{\substack{r_1, \dots, r_d=1 \\ r_{d+1}=r_1}}^{R', \dots, R'} \prod_{k=1}^{d_l} C_k^l(r_k, i_k, r_{k+1})\right) \\ &= (R' \times B \times \sigma_B^2 \times \sigma_\alpha^2)^{d_l} = \sigma_W^2 \end{aligned}$$

[0138] According the equations given here above, it can be noted that the variance of basis tensors is tied up with variances of coefficients matrices elements across all layers. Thus,  $\sigma_B$  is chosen so as to be a function of B and R', the rest of the variance being supported by  $\sigma_\alpha$ . More precisely,  $\sigma_B$  and  $\sigma_\alpha$  are chosen as follows:

$$\sigma_B^2 = (BR')^{-1}, \quad \sigma_\alpha^2 = (\sigma_W)^{2/d_l}$$

internal coefficients  $\alpha_{k,b}^l$  and basis tensors  $\beta_b$  being initialized according to these choices.

[0139] When the diagonal matrices  $\Omega_k^l$  are taken into account in the determination of the reshaped envelope tensor  $\hat{W}_l$  (see step E40 above), said diagonal matrices  $\Omega_k^l$  may be initialized with identity matrices.

[0140] It should be noted that no limitation is attached to the way in which basic tensors, coefficients defining the combinations involving the cores, and, where applicable, diagonal matrices, are initialized. Any method known to the person skilled in the art can therefore be implemented, so that this point is not discussed further here.

**[0141]** Once the first convolutional neural network has been compressed and initialized, the data processing method comprises a step F30 of training said first compressed network, the basis tensors and the coefficients defining said combinations being considered as trainable parameters during said training.

**[0142]** In a still more particular embodiment, when a compression step F10 is carried out such that diagonal matrices are used in the sequence associated with a reshaped envelope tensor of said first convolutional neural network, said diagonal matrices are also considered as learnable parameters during the training of this convolutional neural network.

**[0143]** This training step F20 is performed by the training module MOD\_TRAIN of the data processing system 10.

**[0144]** According to said particular embodiment, the data processing method comprises also a step F40 of compressing a second convolutional neural network, so as to obtain a second compressed network.

**[0145]** This step F40 is performed by the data processing system 10 according to the compression method described above. Moreover, the compression of said second convolutional neural network is implemented using at least the basis tensors learned during the training of the first compressed network.

**[0146]** Thus, the tensor basis generated at the end of the training of the first compressed network are transferred to said second convolutional neural network so that it is compressed using these parameters. In other words, the basis generated for the first compressed neural network serves as a support for the compression of said second convolutional neural network. This way of proceeding is very advantageous because it enables to limit the operating cost in terms of calculation.

**[0147]** It should be noted that the coefficients defining the combinations used to compress the first neural network, as well as the diagonal matrices, may also be transferred between the first and second compressed networks. However, it is understood that these provisions can only be implemented if the first and second networks have very similar structures.

**[0148]** Once the second convolutional neural network has been compressed, the data processing method comprises a step F50 of training said second compressed network, at least coefficients defining the combinations from which said second compressed network is parameterized being considered as trainable parameters during the training of the second compressed network.

**[0149]** According to a first example, only the coefficients of said combinations are considered as trainable parameters during the training of the second compressed network.

**[0150]** According to a second example, not only the coefficients of said combinations but also the basis tensors learned during the training of the first convolutional neural network are considered as trainable parameters during the training of the second compressed network.

**[0151]** In combination with said first and second examples, it may also be considered that when a compression step F40 is carried out such that diagonal matrices are used in the sequence associated with a reshaped envelope tensor of said second convolutional neural network, said diagonal matrices are also considered as learnable parameters during the training of this convolutional neural network.

**[0152]** It should be noted that re-training the basis tensors obtained from training the first compressed network, in addition to the coefficients of the combinations also obtained from training the first compressed network (and also the diagonal matrices, if any), increases the final accuracy of the second compressed network for inference.

**[0153]** Finally, it should also be noted that the data processing process is not limited to the compression of a single second neural network.

**[0154]** The data processing method has been described so far considering that only one second convolutional neural network was compressed during step F40 and then trained during step F50. However, the invention remains of course applicable independently of the number of others convolutional neural networks compressed during step F40 using the basis tensors and the coefficients of the combinations learned during the training of the first compressed network, and also trained during step F50 such that at least the coefficients of said combinations learned during the training of the first compressed network are considered as trainable parameters during the training of another compressed network.

**[0155]** However, the invention also enables to compress and train multiple networks of convolutional neural networks isolated from each other and sharing the same base and the same coefficients trained from a first network of convolutional neurons. By "convolutional neural networks isolated from each other", one refers here to networks assigned to distinct tasks. Therefore, in the case where one of the networks associated with one of these tasks needs to be re-trained, only the coefficients of the combinations (and possibly the basis tensors and diagonal matrices) of this network can be concerned by the re-training. Proceeding in this way allows advantageously not to modify the performances of the networks associated with the other tasks (i.e. one avoids the disadvantage of a negative transfer in the multitask setting).

**[0156]** However, it is also possible to re-train all said other convolutional neural networks at the same time by means of an adapted protocol.

**[0157]** Figure 5 is a graph illustrating performance results obtained by convolutional neural networks compressed according to the invention.

**[0158]** Said performance refers to the Top-1 accuracy (ordinate of the graph in Figure 5) for classification experiments with LeNet5 neural network (see reference [9]) and with MNIST dataset. Classification experiments have been implemented in PyTorch and configured to fit entirely into one GPU with 12 GB.

**[0159]** Model compression is evaluated with the ratio of the number of independent parameters of a compressed neural network (said independent parameters comprise uncompressed parameters which are, for this example, those of the biases and BatchNorm layers as well as those of the very first convolutional layer due to its small size) to the number of independent parameters of the corresponding uncompressed neural network (line with circles annotated with

base size B and rank R, and computed with  $N = 4$ ).  
**[0160]** Model compression is also evaluated with the ratio of the number of independent parameters of a compressed neural network without taking into account the basis tensors, to the number of independent parameters of the corresponding uncompressed neural network (line with crosses annotated with base size B and rank R, and computed with  $N = 4$ ). This value gives an idea of the base-coefficients allocation trade-off and provides an estimate of the number of added parameters in a multitask environment with a shared basis.

**[0161]** In all the experiments, incompressible parameters are trained with the same optimizer settings as prescribed by the baseline training protocol. All other parameters are trained using Adam optimizer in classification tasks with a learning rate equal to 0,003 (see reference [8]). In all experiments, linear LR warm-up for 2000 steps and gradient clipping with L2 norm at 5.0 are performed.

**[0162]** Moreover, the training protocol followed for the experiments of Figure 5 is described in details the reference [4]: 20 epochs, batch size 128, network architecture with two convolutional layers with 20 and 50 output channels respectively, and two linear layers with 320 and 10 output channels, a total of 429K in uncompressed parameters.

**[0163]** As shown in Figure 5, the performance results obtained by the invention, whether or not taking into account the basis tensors for evaluating the compression ratio, surpass the performance results obtained by wide compression method (line with squares).

#### References:

#### **[0164]**

[1] "Deep residual learning for image recognition", He K., Zhang X., Ren S., Sun J., Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778, 2016.

[2] "Very deep convolutional networks for large-scale image recognition", Simonyan K., Zisserman A., International Conference on Learning Representations, 2015.

[3] "Tensor ring decomposition", Zhao Q., Zhou G., Xie S., Zhang L., Cichocki A., arXiv: 1606.05535, 2016.

[4] "Wide compression: Tensor ring nets", Wang W., Sun Y., Eriksson B., Aggarwal V., Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 9329-9338, 2018.

[5] "Learning Filter Basis for Convolutional Neural Network Compression", Li Y., Gu S., Gool L. V., & Timofte R., Proceedings of the IEEE International Conference on Computer Vision, pp. 5623-5632, 2019.

[6] "Tensorizing Neural Networks", Novikov A., Podoprikin D., Osokin A., Vetrov D., Advances in Neural Information Processing Systems, pp. 442-450, 2015.

[7] "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification", He K., Zhang X., Ren S., Sun J., IEEE international conference on computer vision, pp. 1026-1034, 2015.

[8] "Adam: A method for stochastic optimization", Kingma D., Ba J., Arxiv: 1412.6980, 2015.

[9] "Gradient-based learning applied to document recognition", Lecun Y., Bottou L., Bengio Y., Haffner P., Proceedings of the IEEE, 86(11):2278:2324, 1998.

#### Claims

1. A method for compressing a neural network comprising a plurality of layers, each layer comprising a weight tensor whose size is equal to a product of numbers referred to as "size numbers", said method comprising a step of selecting (E10) hyperparameters to define at least one basis of tensors referred to as "basis tensors", including a rank hyperparameter R, a dimension hyperparameter N and a size hyperparameter representative of the number of basis tensors, each basis tensor being an array belonging to a vector space whose dimension is equal to the product of

at least one number belonging to the hyperparameter R and a number N' belonging to the hyperparameter N, said method also comprising, for each weight tensor, a set of steps of:

- embedding (E20) the weight tensor into an envelope tensor, the size of the envelope tensor being equal to a product of size numbers such that:

- if a size number M of the weight tensor is an integer power of said number N', said size number M is a size number of the envelope tensor,
- if a size number M of the weight tensor is not an integer power of said number N', said number N' raised to a power equal to the ceiling value of log base N' of said size number M is a size number of the envelope tensor,

- reshaping (E30) said envelope tensor by tensorization based on said number N',  
 - determining (E40) a representation of the reshaped envelope tensor in the form of a sequence of elements comprising tensors referred to as "cores", said cores belonging to said vector space,  
 - parameterizing (E50) each core into a combination of the basis tensors.

2. The method according to claim 1, wherein, for each weight tensor, the sequence associated with the reshaped envelope tensor also comprises, before each core, a diagonal matrix intended to be multiplied with said core following it in said sequence, the size of a diagonal matrix being defined as a function of the hyperparameter R.

3. The method according to any one of claims 1 to 2, wherein, for each weight tensor, the representation determined for the reshaped envelope tensor is a tensor ring decomposition.

4. The method according to any one of claims 1 to 2, wherein, for each weight tensor, the representation determined for the reshaped envelope tensor is a tensor train decomposition.

5. The method according to any one of claims 1 to 4, wherein said at least one basis is formed of three-dimensional tensors or else four-dimensional tensors.

6. The method according to any one of claims 1 to 5, wherein the basis tensors satisfy at least one constraint parameterized by the number of independent parameters defining each of said tensors.

7. A data processing method comprising the steps of:

- compressing (F10) a first neural network according to any one of claims 1 to 6, so as to obtain a first compressed network,  
 - initializing (F20) the basis tensors and the combinations from which said first compressed network is parameterized,  
 - training (F30) said first compressed network, the basis tensors and coefficients defining said combinations being considered as trainable parameters during said training.

8. The method according to claim 7, said method also comprising the steps of:

- compressing (F40) at least one second neural network according to any one of claims 1 to 6, so as to obtain a second compressed network, the compression of said second neural network being implemented using at least the basis tensors,  
 - training (F50) said second compressed network, at least coefficients defining the combinations from which said second compressed network is parameterized being considered as trainable parameters during the training of the second compressed network.

9. The method according to claim 8, wherein the basis tensors learned during the training of the first neural network are considered as trainable parameters during the training of the second compressed network.

10. A method according to any one of claims 7 to 9, wherein, when a compression step of a neural network is carried out according to claim 2, the diagonal matrices used for this neural network are also initialized during the initialization step and are also considered as learnable parameters during the training of this neural network.



11. A computer program including instructions for executing the compression method according to any one of claims 1 to 6 or a processing data method according to any one of the claims 7 to 10 when said program is executed by a computer.

12. A recording medium readable by a computer and having recorded thereon a computer program according to claim 11.

13. A data processing system for compressing a neural network comprising a plurality of layers, each layer comprising a weight tensor whose size is equal to a product of numbers referred to as "size numbers", said processing system comprising:

- a selection module (MOD\_SEL), configured to select hyperparameters to define at least one basis of tensors referred to as "basis tensors", including a rank hyperparameter R, a dimension hyperparameter N and a size hyperparameter representative of the number of basis tensors, each basis tensor being an array belonging to a vector space whose dimension is equal to the product of at least one number belonging to the hyperparameter R and a number N' belonging to the hyperparameter N,

- an embedding module (MOD\_EMB), configured to embed a weight tensor into an envelope tensor, the size of the envelope tensor being equal to a product of size numbers such that:

- if a size number M of the weight tensor is an integer power of said number N', said size number M is a size number of the envelope tensor,
- if a size number M of the weight tensor is not an integer power of said number N', said number N' raised to a power equal to the ceiling value of log base N' of said size number M is a size number of the envelope tensor,

- a reshaping module (MOD\_RSHA), configured to reshape an envelope tensor by tensorization based on said number N',

- a determination module (MOD\_DET), configured to determine, for a reshaped envelope tensor, a representation of said reshaped envelope tensor in the form of a sequence of elements comprising tensors referred to as "cores", said cores belonging to said vector space,

- a parameterization module (MOD\_PAR), configured to parameterize a core into a combination of the basis tensors.

14. A system comprising:

- an initialization module (MOD\_INI), configured to initialize the basis tensors and the combinations from which a neural network is parameterized, said neural network having been compressed by means of a data processing system according to claim 13,

- a training module (MOD\_TRAIN), configured to train said compressed neural network, the basis tensors and coefficients defining said combination being considered as trainable parameters during said training.

Amended claims in accordance with Rule 137(2) EPC.

1. A computer-implemented method for compressing a neural network comprising a plurality of layers, each layer comprising a weight tensor whose size is equal to a product of numbers referred to as "size numbers", said method comprising a step of selecting (E10) hyperparameters to define at least one basis of tensors referred to as "basis tensors", including a rank hyperparameter R, a dimension hyperparameter N and a size hyperparameter representative of the number of basis tensors, each basis tensor being an array belonging to a vector space whose dimension is equal to the product of at least one number belonging to the hyperparameter R and a number N' belonging to the hyperparameter N, said method also comprising, for each weight tensor, a set of steps of:

- embedding (E20) the weight tensor into an envelope tensor, the size of the envelope tensor being equal to a product of size numbers such that:

- if a size number M of the weight tensor is an integer power of said number N', said size number M is a size number of the envelope tensor,
- if a size number M of the weight tensor is not an integer power of said number N', said number N' raised to a power equal to the ceiling value of log base N' of said size number M is a size number of the envelope tensor,

- reshaping (E30) said envelope tensor by tensorization based on said number N',
- determining (E40) a representation of the reshaped envelope tensor in the form of a sequence of elements comprising tensors, said tensors of the sequence being referred to as "cores" and belonging to said vector space,
- parameterizing (E50) each core into a combination of the basis tensors.

2. The method according to claim 1, wherein, for each weight tensor, the sequence associated with the reshaped envelope tensor also comprises, before each core, a diagonal matrix intended to be multiplied with said core following it in said sequence, the size of a diagonal matrix being defined as a function of the hyperparameter R.

3. The method according to any one of claims 1 to 2, wherein, for each weight tensor, the representation determined for the reshaped envelope tensor is a tensor ring decomposition.

4. The method according to any one of claims 1 to 2, wherein, for each weight tensor, the representation determined for the reshaped envelope tensor is a tensor train decomposition.

5. The method according to any one of claims 1 to 4, wherein said at least one basis is formed of three-dimensional tensors or else four-dimensional tensors.

6. The method according to any one of claims 1 to 5, wherein the basis tensors satisfy at least one constraint parameterized by the number of independent parameters defining each of said basis tensors.

7. A data processing method comprising the steps of:

- compressing (F10) a first neural network according to any one of claims 1 to 6, so as to obtain a first compressed network,
- initializing (F20) the basis tensors and the combinations from which said first compressed network is parameterized,
- training (F30) said first compressed network, the basis tensors and coefficients defining said combinations being considered as trainable parameters during said training.

8. The method according to claim 7, said method also comprising the steps of:

- compressing (F40) at least one second neural network according to any one of claims 1 to 6, so as to obtain a second compressed network, the compression of said second neural network being implemented using at least the basis tensors,
- training (F50) said second compressed network, at least coefficients defining the combinations from which said second compressed network is parameterized being considered as trainable parameters during the training of the second compressed network.

9. The method according to claim 8, wherein the basis tensors learned during the training of the first neural network are considered as trainable parameters during the training of the second compressed network.

10. A method according to any one of claims 7 to 9, wherein, when a compression step of a neural network is carried out according to claim 2, the diagonal matrices used for this neural network are also initialized during the initialization step and are also considered as learnable parameters during the training of this neural network.

11. A computer program including instructions for executing the compression method according to any one of claims 1 to 6 or a processing data method according to any one of the claims 7 to 10 when said program is executed by a computer.

12. A recording medium readable by a computer and having recorded thereon a computer program according to claim 11.

13. A data processing system for compressing a neural network comprising a plurality of layers, each layer comprising a weight tensor whose size is equal to a product of numbers referred to as "size numbers", said processing system comprising:

- a selection module (MOD\_SEL), configured to select hyperparameters to define at least one basis of tensors referred to as "basis tensors", including a rank hyperparameter R, a dimension hyperparameter N and a size

hyperparameter representative of the number of basis tensors, each basis tensor being an array belonging to a vector space whose dimension is equal to the product of at least one number belonging to the hyperparameter R and a number N' belonging to the hyperparameter N,

- an embedding module (MOD\_EMB), configured to embed a weight tensor into an envelope tensor, the size of the envelope tensor being equal to a product of size numbers such that:

- if a size number M of the weight tensor is an integer power of said number N', said size number M is a size number of the envelope tensor,
- if a size number M of the weight tensor is not an integer power of said number N', said number N' raised to a power equal to the ceiling value of log base N' of said size number M is a size number of the envelope tensor,

- a reshaping module (MOD\_RSHA), configured to reshape an envelope tensor by tensorization based on said number N',

- a determination module (MOD\_DET), configured to determine, for a reshaped envelope tensor, a representation of said reshaped envelope tensor in the form of a sequence of elements comprising tensors, said tensors of the sequence being referred to as "cores" and belonging to said vector space,

- a parameterization module (MOD\_PAR), configured to parameterize a core into a combination of the basis tensors.

#### 14. A system comprising:

- an initialization module (MOD\_INI), configured to initialize the basis tensors and the combinations from which a neural network is parameterized, said neural network having been compressed by means of a data processing system according to claim 13,

- a training module (MOD\_TRAIN), configured to train said compressed neural network, the basis tensors and coefficients defining said combination being considered as trainable parameters during said training.

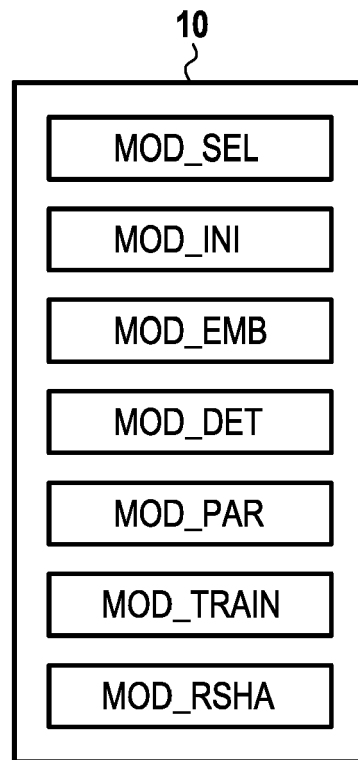


FIG.1

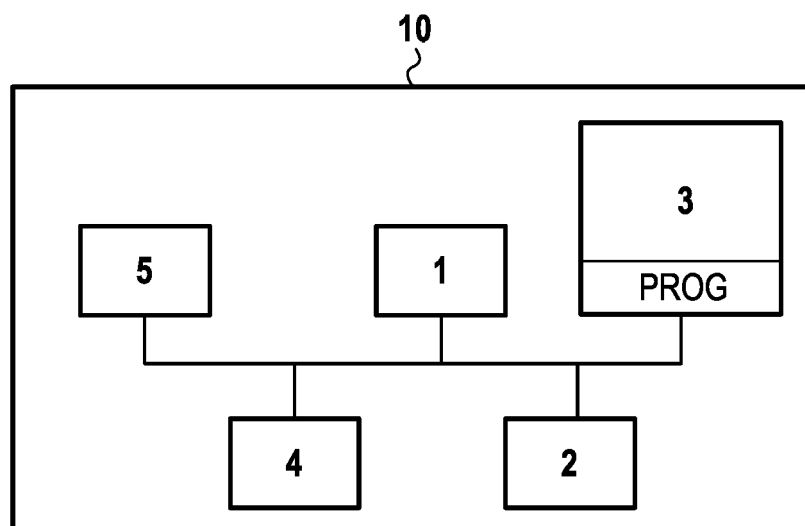


FIG.2

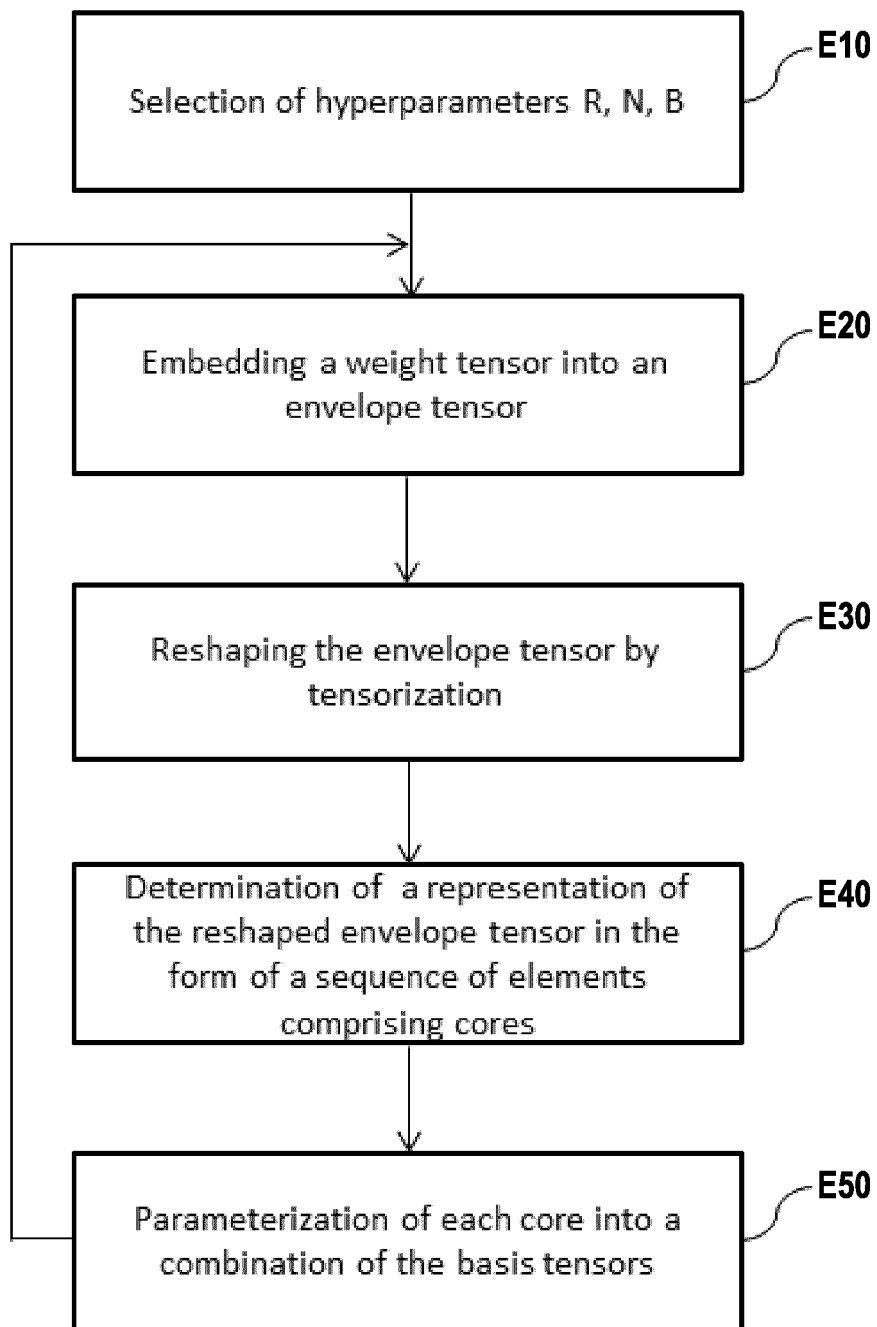


FIG.3

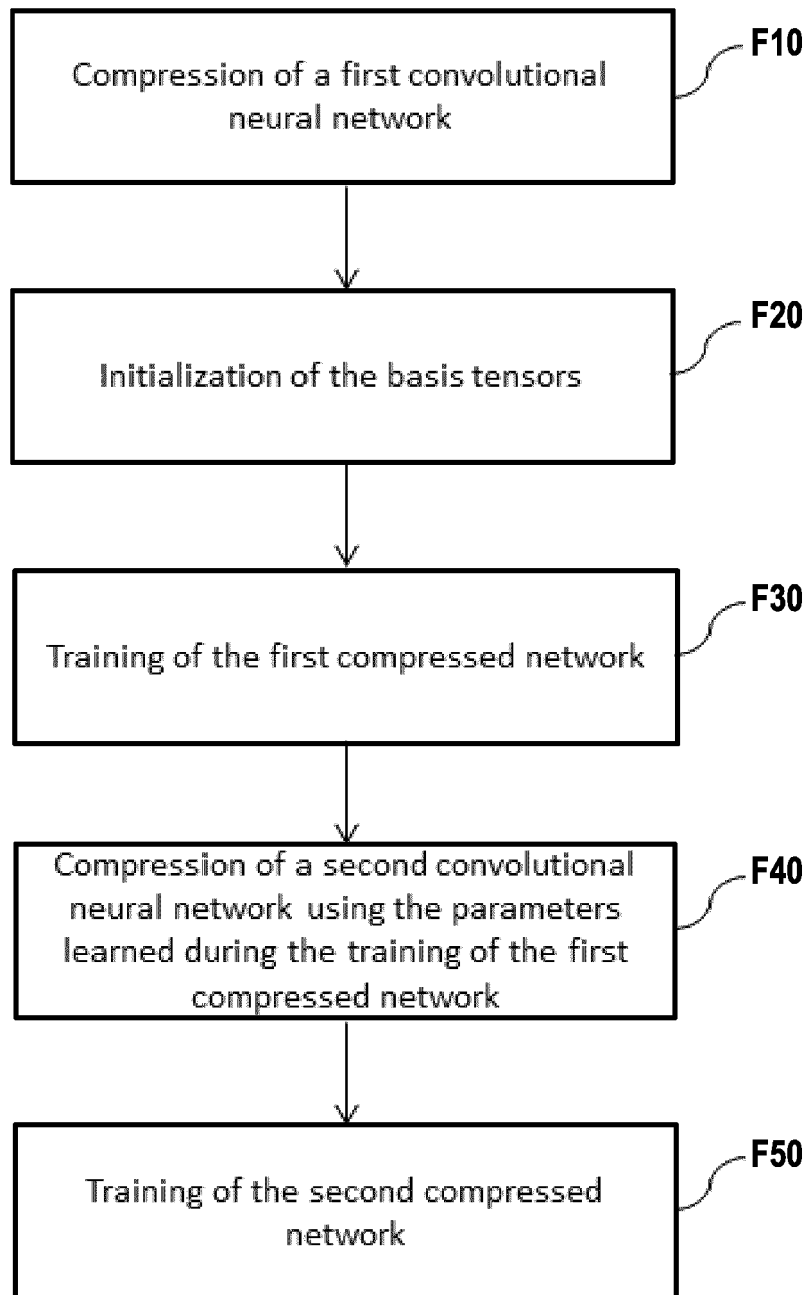


FIG.4

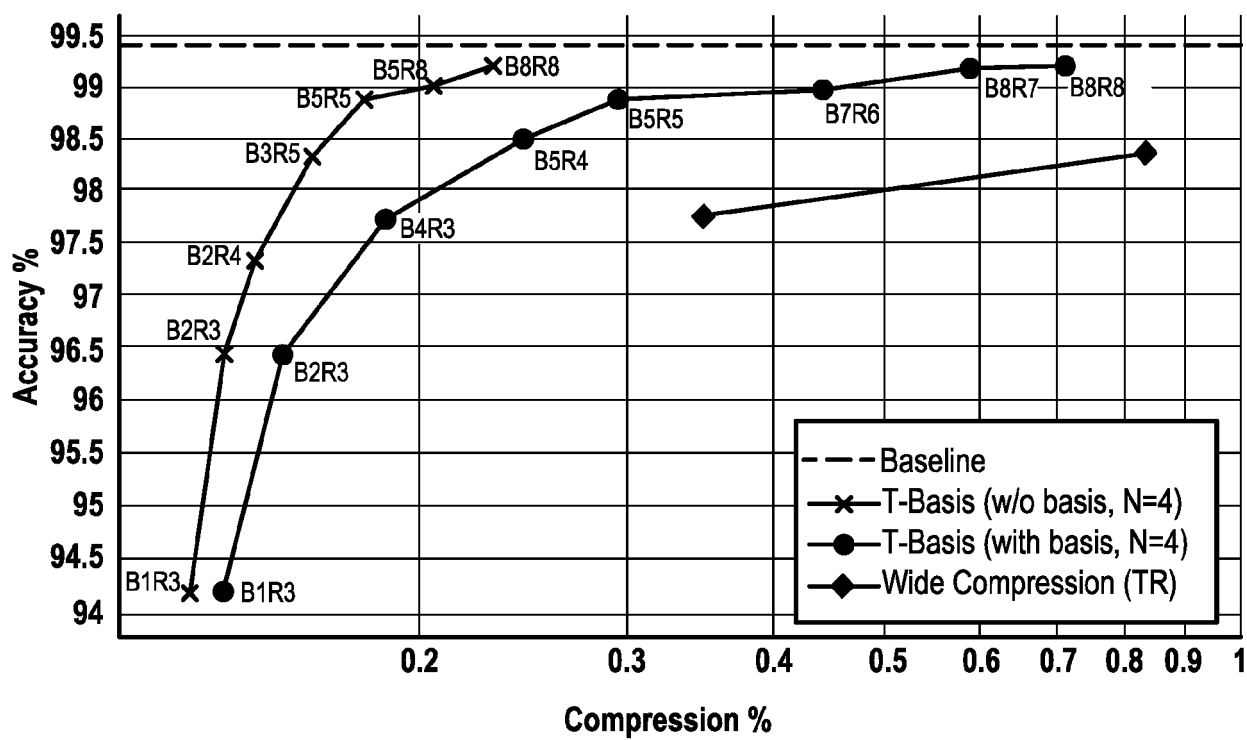


FIG.5



## EUROPEAN SEARCH REPORT

Application Number  
EP 20 17 9099

5

10

15

20

25

30

35

40

45

50

55

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (IPC)
X,D	AGGARWAL VANEET ET AL: "Wide Compression: Tensor Ring Nets", 2018 IEEE/CVF CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION, IEEE, 18 June 2018 (2018-06-18), pages 9329-9338, XP033473859, DOI: 10.1109/CVPR.2018.00972 [retrieved on 2018-12-14] * the whole document *	1-14	INV. G06N3/04 G06N3/08
A	LI YAWEI ET AL: "Learning Filter Basis for Convolutional Neural Network Compression", 2019 IEEE/CVF INTERNATIONAL CONFERENCE ON COMPUTER VISION (ICCV), IEEE, 27 October 2019 (2019-10-27), pages 5622-5631, XP033723354, DOI: 10.1109/ICCV.2019.00572 [retrieved on 2020-02-24] * the whole document *	1-14	TECHNICAL FIELDS SEARCHED (IPC) G06N
The present search report has been drawn up for all claims			
Place of search The Hague		Date of completion of the search 30 November 2020	Examiner Jacobs, Jan-Pieter
CATEGORY OF CITED DOCUMENTS X : particularly relevant if taken alone Y : particularly relevant if combined with another document of the same category A : technological background O : non-written disclosure P : intermediate document		T : theory or principle underlying the invention E : earlier patent document, but published on, or after the filing date D : document cited in the application L : document cited for other reasons & : member of the same patent family, corresponding document	

 1  
EPO FORM 1503 03.82 (P04C01)



## REFERENCES CITED IN THE DESCRIPTION

*This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.*

## Non-patent literature cited in the description

- **HE K. ; ZHANG X. ; REN S. ; SUN J.** Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, 770-778 [0164]
- **SIMONYAN K. ; ZISSERMAN A.** Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations*, 2015 [0164]
- **ZHAO Q. ; ZHOU G. ; XIE S. ; ZHANG L. ; CICHOCKI A.** Tensor ring decomposition, 2016 [0164]
- **WANG W. ; SUN Y. ; ERIKSSON B. ; AGGARWAL V.** Wide compression: Tensor ring nets. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, 9329-9338 [0164]
- **LI Y. ; GU S. ; GOOL L. V. ; TIMOFTE R.** Learning Filter Basis for Convolutional Neural Network Compression. *Proceedings of the IEEE International Conference on Computer Vision*, 2019, 5623-5632 [0164]
- **NOVIKOV A. ; PODOPRIKHIN D. ; OSOKIN A. ; VETROV D.** Tensorizing Neural Networks. *Advances in Neural Information Processing Systems*, 2015, 442-450 [0164]
- **HE K. ; ZHANG X. ; REN S. ; SUN J.** Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *IEEE international conference on computer vision*, 2015, 1026-1034 [0164]
- **KINGMA D. ; BA J.** Adam: A method for stochastic optimization, 2015 [0164]
- **LECUN Y. ; BOTTOU L. ; BENGIO Y. ; HAFFNER P.** Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998, vol. 86 (11), 2278-2324 [0164]